Answer categorization in a question answering system

Anders Tilly
D10, Lund Institute of Technology, Sweden
ada09ati@student.lu.se

January 18, 2014

Abstract
This paper describes the process of categorizing answers to questions in a corpus of questions for use in a question answering system. The focus is on the actual classification of the questions themselves. A comparison of methods is made, and an update is made to the corpus for further testing purposes.

Introduction

The basic idea behind a question-answering system is very simple: A question is sent to the system as input and it produces an answer to the question as output. This project was inspired by the IBM Watson, a question answering system developed by IBM[1]. IBM Watson began development in 2005 with the goal of being able to defeat human opponents in Jeopardy, which it successfully did in 2011[2]. In particular, this paper focuses on question analysis and answer categorization. The question analysis takes a question and tries to predict which category the answer to the question is in. Categorization was made with the help of Libshottext, a tool for short-text classification[3].

Method

The initial corpus consisted of 2079 questions, taken from a Swedish trivia game called "kvitt eller dubbelt", a game similar to Jeopardy. Of the questions, all of them had answers but only some of them had answer categories. Libshorttext was initially trained on a training set consisting of 90% of the questions and tested on a test set consisting of 10% of the question. When tested this way with the questions having initial answer categories, the answer category was correctly predicted 82,7% of the time. This can be considered a fairly good result. However, since the corpus was incomplete, an attempt was made at manually updating the missing categories in the corpus. The different categories were: human, location, entity, numeric, abrev, description and action. When the corpus was updated an additional category was added, work, used to describe creative works such as movies, songs and others.

When tested on a new training set and test set based on the new updated corpus, the answer category was correctly predicted 79,2% of the time. This is actually worse than the initial result, however, this is understandable when you consider the fact that the new category, work, was not applied on a particularly large amount of questions and thus it might not actually have helped libshorttext in correctly predicting the category.

In addition to this, some tweaking can be made to libshorttext itself. It comes with a set of parameters that can be adjusted to improve performance. The first parameter is -P, and it controls preprocessor options, and the options are whether to have stopword removal,

stemming, and whether to use bigrams or unigrams. The default option is to use no stopword removal, no stemming, and bigrams, and it was also the option that produced the best result. The second parameter, -G, can be either 0 or 1 and decides whether to use a grid search for the linear classifiers. Default is 0, however, setting it at 1 did not produce better results so it was left at 0 as prediction is faster this way.

The third parameter is -F, and it decides which feature representation to use. The available ones are binary feature, word count, term frequency and TF-IDF(Term Frequency + Inverse Document Frequency). The default here is binary representation, but it was not used as TF-IDF produced better results. The fourth parameter is -N, and it can be either 0 or 1 and it decides whether or not to do instant-wise normalization before training/test. The results were better with normalization so it was left at 1, which is the default value. The fifth parameter is -L, which decides which classifier to use. The different options are support vector classification by Crammer and Singer, L1-loss support vector classification, L2-loss support vector classification and logistic regression. The best result was achieved with the default value, vector classification by Crammer and Singer, though in some instances one might want to use logistic regression to compute probabilities for the different categories. Therefore, for the best result possible, the default values were used except for the feature representation where TF-IDF was better. The final results were 83.6% accuracy for the old set and 81.4% for the new set.

Related work

There are numerous articles about the development of IBM Watson the reader might be interested in, one of which is cited in the References section.

Conclusions

Even though the result was actually worse with the new corpus, it could still be used since the additional category might be more helpful in some situations. Libshorttext states that its default values are carefully selected, and while they were mostly the best ones, TF-IDF was used for the feature representation to achieve the best result for questions from this corpus.

Acknowledgements

This project was worked on under the supervision of Pierre Nugues, and was in part collaberated with Jakob Grundström.

References

[1] J. Chu-Carrol, J. Fan, B. K. Boguraev, D. Carmel, D. Sheinwald, and C. Welty, *Finding needles in the haystack: Search and candidate generation*

[2] Baker, Stephen (2011). *Final Jeopardy: Man vs. Machine and the Quest to Know Everything*. Boston, New York: Houghton Mifflin Harcourt

[3] Hsiang-Fu Yu, Chia-Hua Ho, Yu-Chin Juan, Chih-Jen Lin, *LibShortText: A Library for Short-text Classification and Analysis*